arVix

December 1, 2021

1 ArXiv

[1]:
```
!export PATH=/Library/TeX/texbin:/Library/TeX/texbin/xelatex
import random
import math
import numpy as np
import pandas as pd
import time
```

1.0.1 Readfile Functions

[2]:
```
#input filename(str)
#output file(list)
def readfile(filename):
    with open(filename) as file_in:
        lines = []
        for line in file_in:
            lines.append(line)
    return lines
```

[3]:
```
import json
#input file(list)
#output json(list of dic)
def list2json(lines):
    jsons = []
    for i in range(len(lines)):
        tmp = json.loads(lines[i])
        jsons.append(tmp)
    return jsons
```

1.0.2 Calculus Functions

[4]:
```
# input: article
# output: number
def avg_token(article):
    s = 0
    for sentence in article:
        token = sentence.split()
        n = len(token)
        s += n
    return s/len(article)
```

[5]:
```
def total_token(article):
    s = 0
    for sentence in article:
        token = sentence.split()
        n = len(token)
        s += n
    return s
```

[6]:
```
#input (list of dic)
#output numbers
def min_max_avg(jsons, name, type_no):
    Min = len(jsons[0][name])
    Max = len(jsons[0][name])
    short_article=jsons[0][name]
    SUM = 0
    numbers = []

    for i in range(len(jsons)):
        article = jsons[i][name]
        if type_no == 1:
            N = avg_token(article)
        elif type_no == 2:
            N = total_token(article)
        else:
            N = len(article)

        numbers.append(N)
        SUM += N
        if Min>N:
            Min = N
            #short_article=jsons[i][name]
        if Max<N:
            Max = N

    Avg = SUM//len(jsons)
    #print(short_article)
    return (Min, Max, Avg, len(jsons), numbers)
```

1.0.3 Print Output Functions

[7]:
```
from matplotlib import pyplot as plt

def plot_graph(bin_list, numbers, image_name):
    plt.hist(numbers, bins = bins_list)
    plt.savefig(image_name)
    plt.show()
```

[8]:
```
def print_result(title, Min, Max, Avg, l):
    #-------------------------------
    print(title)
    print('------------------------------------------------------------')
    print('Number of articles:'+str(l))
    print('Longest:'+str(Max))
    print('Shortest:'+str(Min))
    print('Average:'+str(Avg))
```

[9]:
```
def print_out(jsons, name_str, output_str, type_no):
    Min, Max, Avg, l, numbers = min_max_avg(jsons, name_str, type_no)
    print_result(output_str, Min, Max, Avg, l)
    return numbers
```

1.1 Test data

[10]:
```
test = readfile('arxiv-dataset/test.txt')
test_jsons= list2json(test)
```

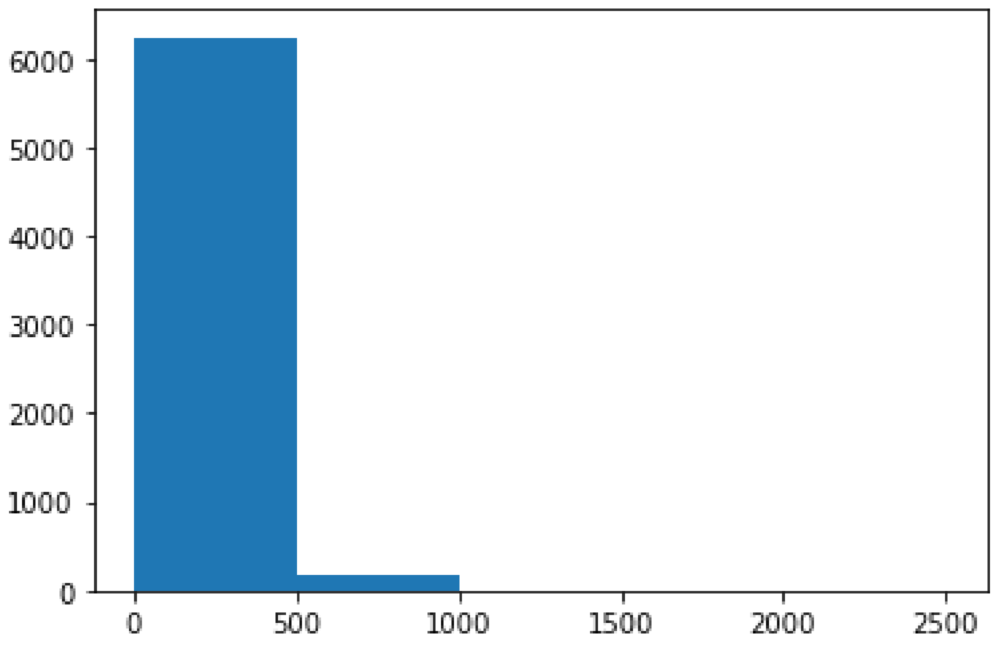
1.1.1 Test data: number of sentences in an article

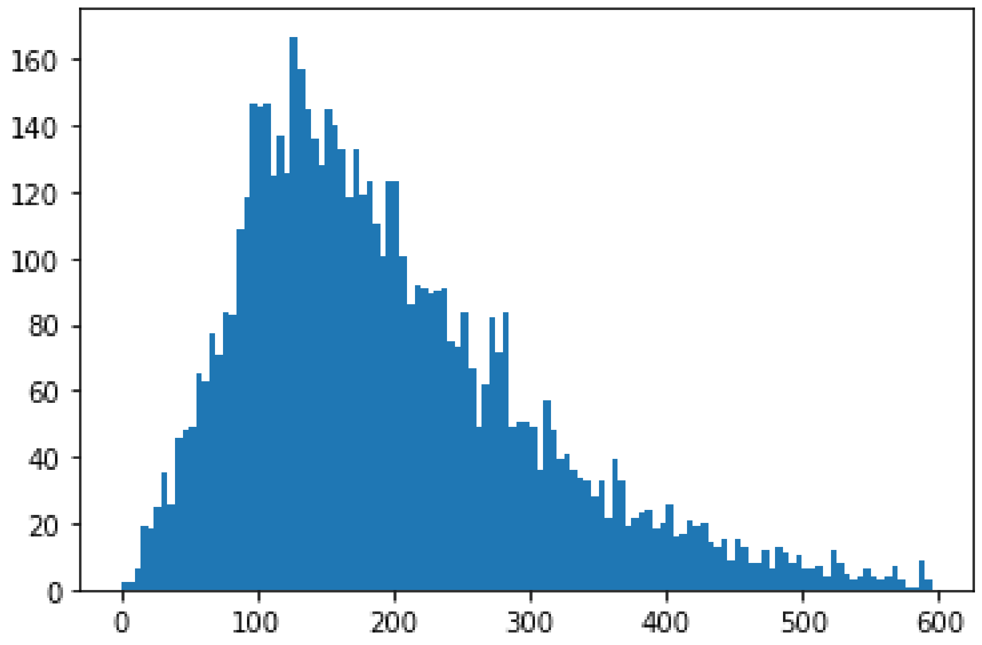
[11]:
```
test_s_numbers = print_out(test_jsons, 'article_text', 'Test Data', 3)
```

```
Test Data
------------------------------------------------------------
Number of articles:6440
Longest:3045
Shortest:1
Average:205
```

[12]:
```
bins_list = list(range(0,3000,500))
plot_graph(bins_list, test_s_numbers, 'test_s_1.png')

bins_list = list(range(0,600,5))
plot_graph(bins_list, test_s_numbers, 'test_s_2.png')
```





Test data: number of tokens in a sentence

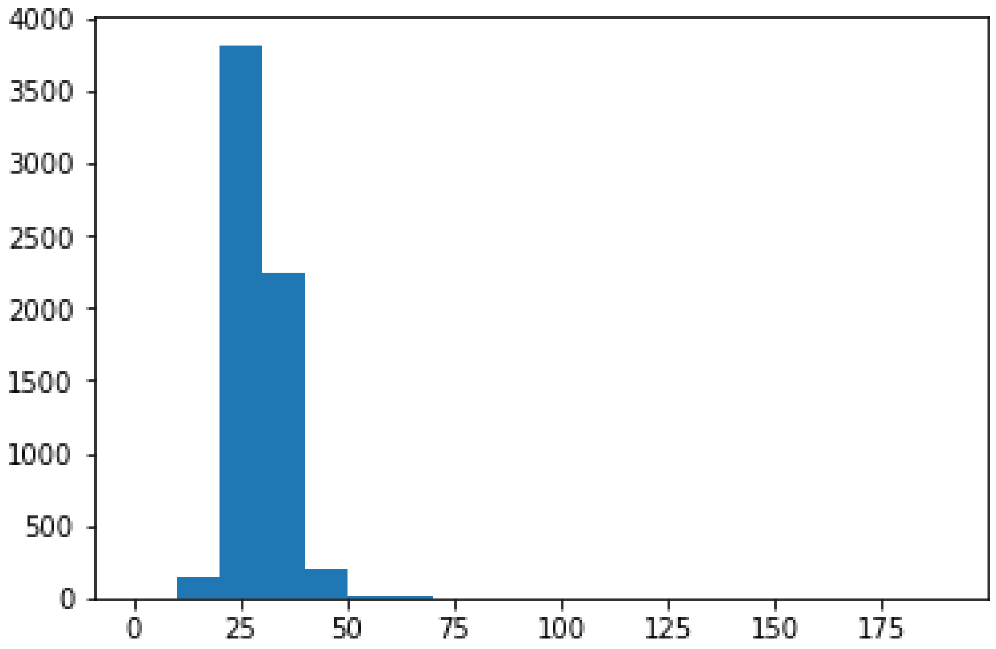
[13]:
```
test_t1_numbers = print_out(test_jsons, 'article_text', 'Test Data', 1)
```

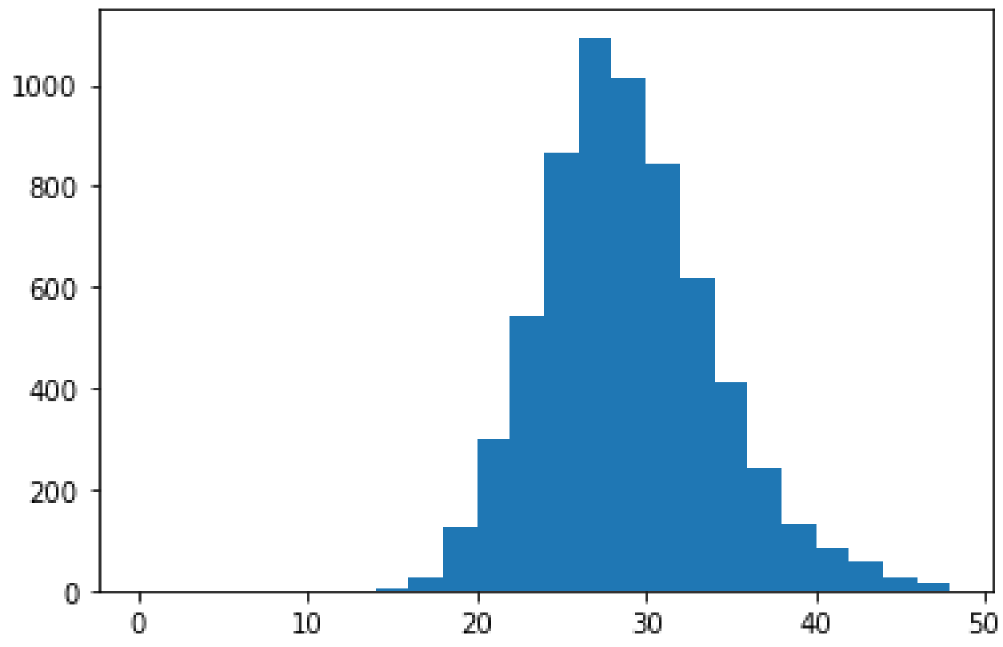
```
Test Data
```

```
-------------------------------------------------------------
Number of articles:6440
Longest:220
Shortest:14.373333333333333
Average:29.0
```

[14]:
```
bins_list = list(range(0, 200, 10))
plot_graph(bins_list, test_t1_numbers, 'test_t1_1.png')

bins_list = list(range(0, 50, 2))
plot_graph(bins_list, test_t1_numbers, 'test_t1_2.png')
```





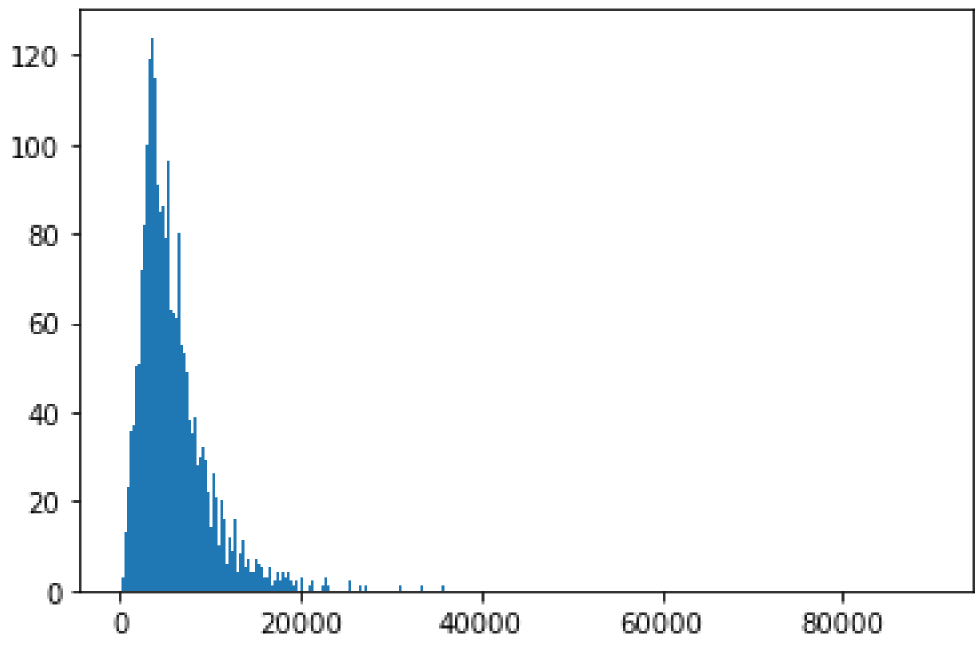
1.1.2 Test data: number of tokens in an article

```
[15]: test_t2_numbers = print_out(test_jsons, 'article_text', 'Test Data', 2)
```

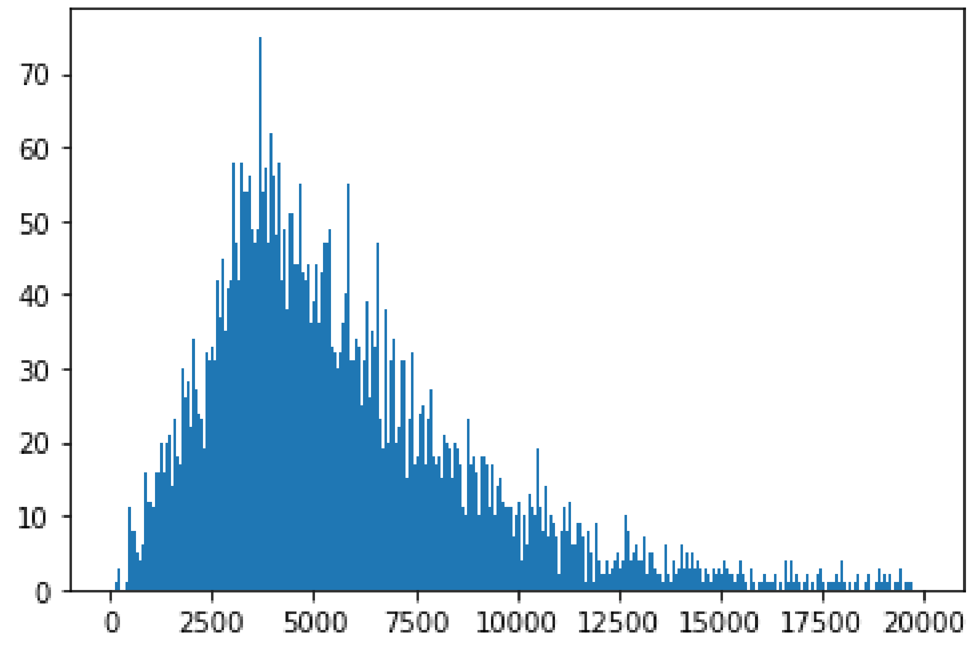
```
Test Data
-------------------------------------------------------------
Number of articles:6440
Longest:84895
Shortest:105
Average:5905
```

```
[16]: bins_list = list(range(100,90000,100))
      plot_graph(bins_list, test_t2_numbers, 'test_t2_1.png')

      bins_list = list(range(0,20000,50))
      plot_graph(bins_list, test_t2_numbers, 'test_t2_2.png')
```



120
100
80
60
40
20
0
0
20000
40000
60000
80000



70
60
50
40
30
20
10
0
0
2500
5000
7500
10000
12500
15000
17500
20000

1.2 Train data

[17]:
```
start = time.time()
#---------------------------------------------------------------------------
train = readfile('arxiv-dataset/train.txt')
train_jsons= list2json(train)
#---------------------------------------------------------------------------
print(time.time()-start)
```

```
889.1048378944397
```

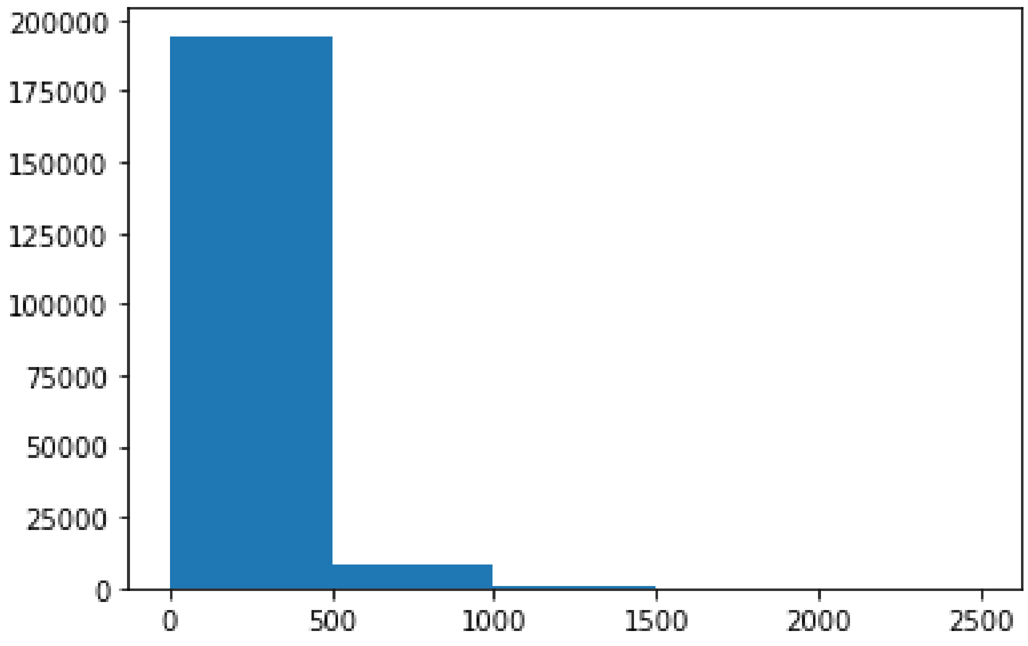
1.2.1 Train data: number of sentences in an article

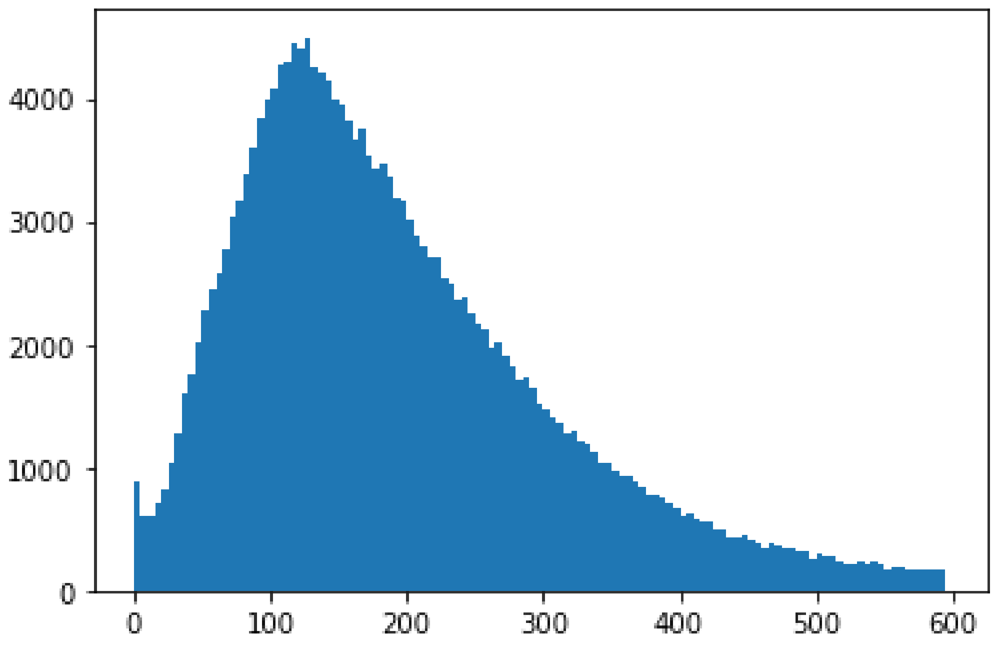
[18]:
```
train_s_numbers = print_out(train_jsons, 'article_text', 'Train Data', 3)
```

```
Train Data
-------------------------------------------------------------
Number of articles:203037
Longest:4615
Shortest:1
Average:206
```

[19]:
```
bins_list = list(range(0,3000,500))
plot_graph(bins_list, train_s_numbers, 'train_s_1.png')

bins_list = list(range(0,600,5))
plot_graph(bins_list, train_s_numbers, 'train_s_2.png')
```



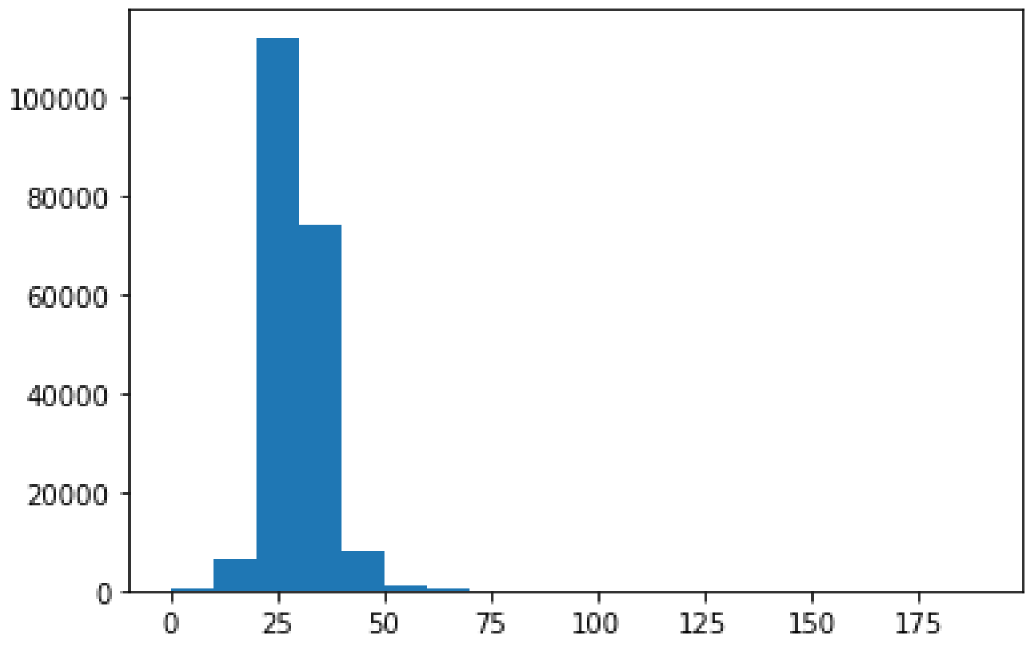


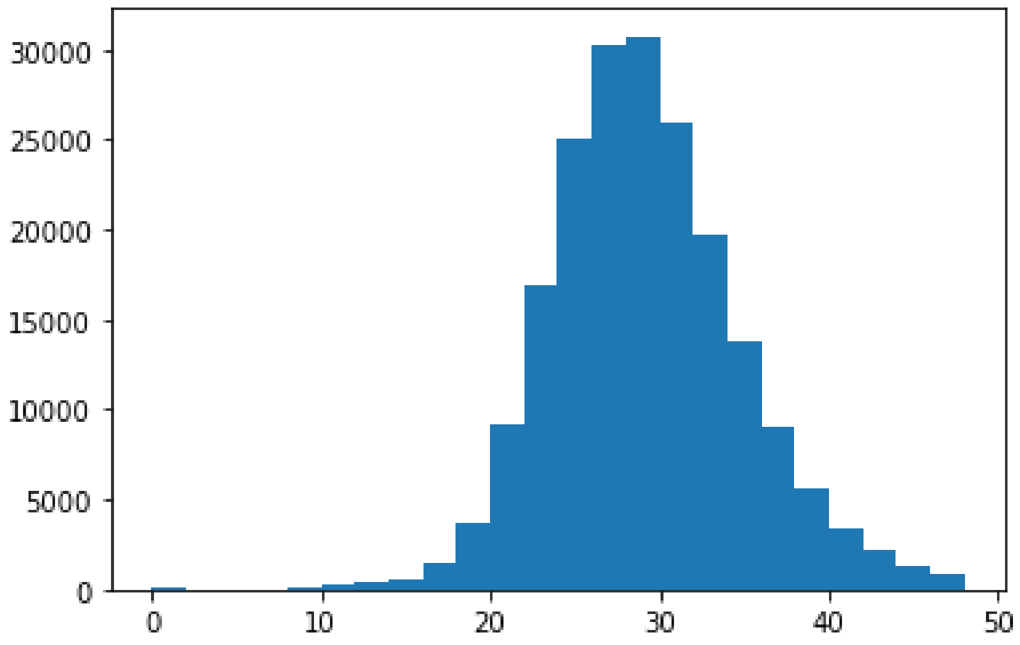
1.2.2 Train data: number of tokens in a sentence

[20]:
```
train_t1_numbers = print_out(train_jsons, 'article_text', 'Train Data', 1)
bins_list = list(range(0, 200, 10))
plot_graph(bins_list, train_t1_numbers, 'train_t1_1.png')

bins_list = list(range(0, 50, 2))
plot_graph(bins_list, train_t1_numbers, 'train_t1_2.png')
```

```
Train Data
-------------------------------------------------------------
Number of articles:203037
Longest:2317.0
Shortest:0.0
Average:29.0
```





1.2.3 Train data: number of tokens in an article

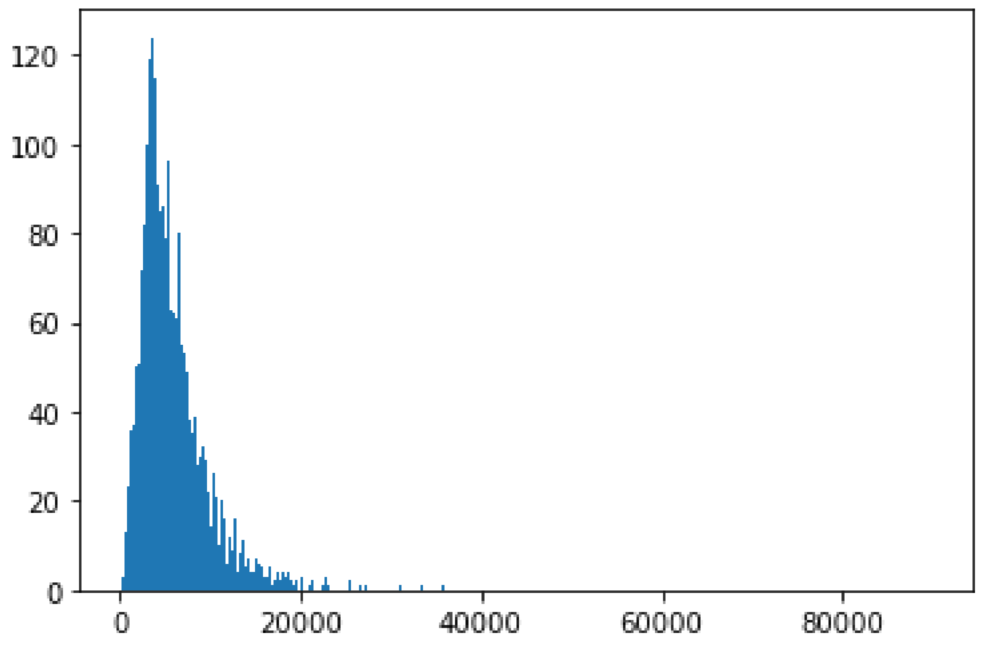
[21]:
```
train_t2_numbers = print_out(test_jsons, 'article_text', 'Train Data', 2)
```

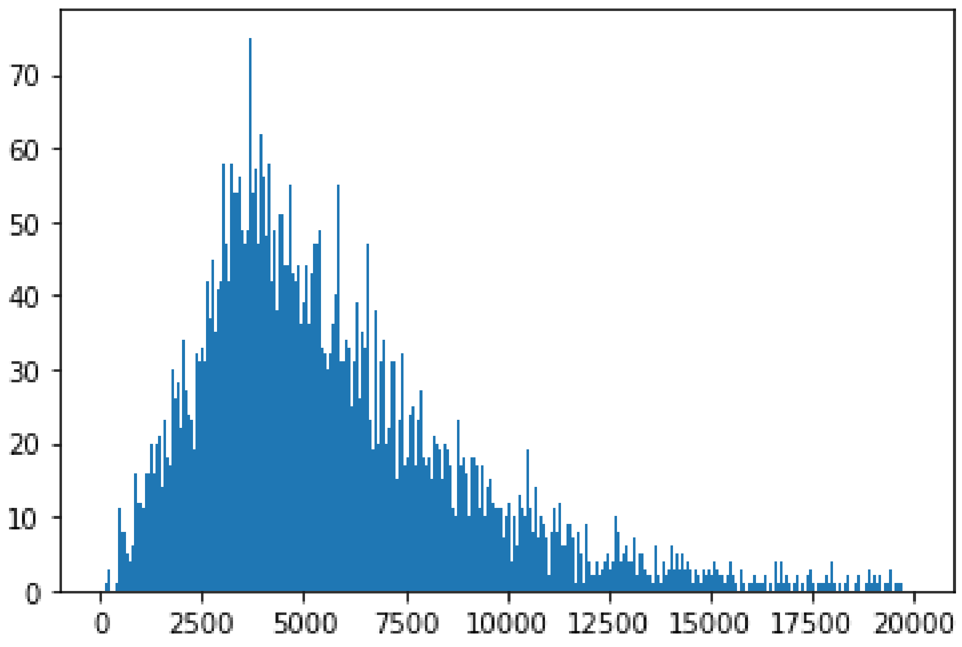
```
Train Data
```

```
------------------------------------------------------------
Number of articles:6440
Longest:84895
Shortest:105
Average:5905
```

[22]:
```
bins_list = list(range(100,90000,100))
plot_graph(bins_list, train_t2_numbers, 'train_t2_1.png')

bins_list = list(range(0,20000,50))
plot_graph(bins_list, train_t2_numbers, 'train_t2_2.png')
```





1.3 Validation

[23]:
```python
start = time.time()
#-------------------------------------------------------------------------
val = readfile('arxiv-dataset/val.txt')
val_jsons= list2json(val)
#-------------------------------------------------------------------------
print(time.time()-start)
```

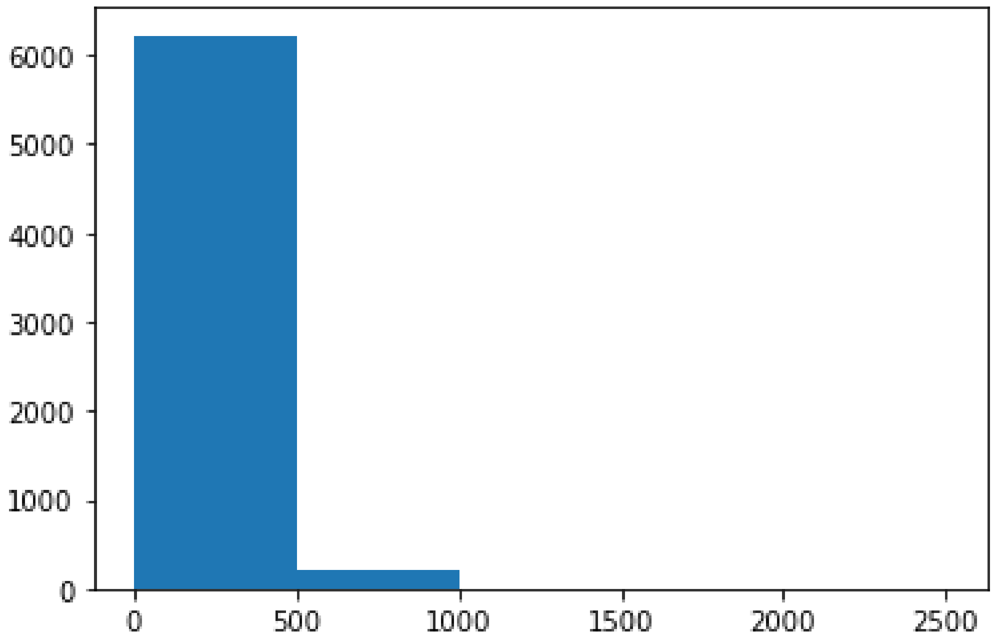
```
4.360097885131836
```

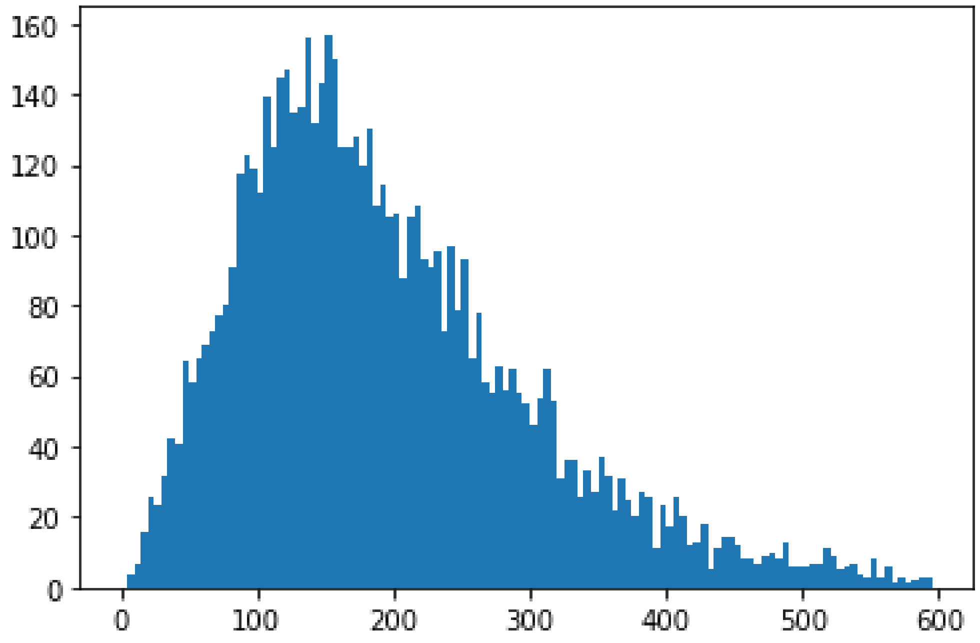
1.3.1 Validation data: number of sentences in an article

[24]:
```python
val_s_numbers = print_out(val_jsons, 'article_text', 'Validation Data', 3)
bins_list = list(range(0, 3000, 500))
plot_graph(bins_list, val_s_numbers, 'val_s_1.png')

bins_list = list(range(0, 600, 5))
plot_graph(bins_list, val_s_numbers, 'val_s_2.png')
```

```
Validation Data
-------------------------------------------------------------
Number of articles:6436
Longest:1483
Shortest:8
Average:204
```


6000
5000
4000
3000
2000
1000
0
0
500
1000
1500
2000
2500

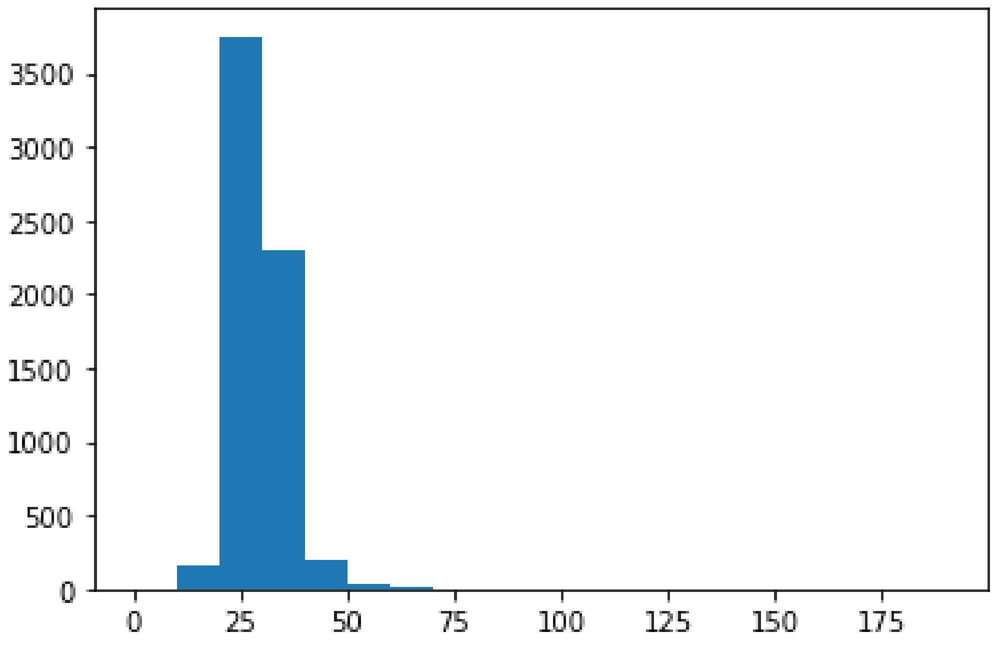

160
140
120
100
80
60
40
20
0
0
100
200
300
400
500
600

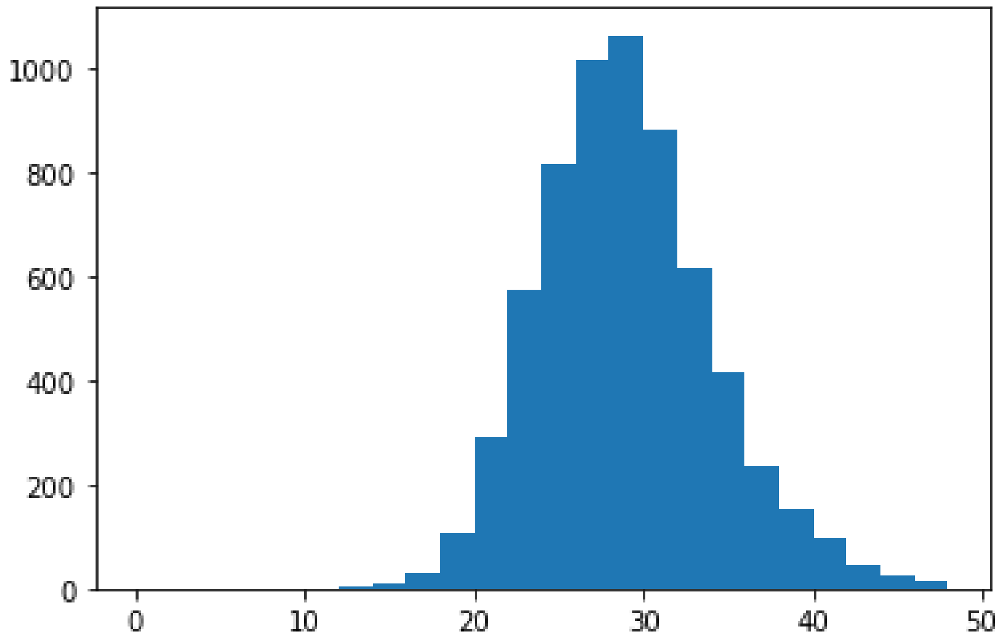
1.3.2 Validation data: number of tokens in a sentence

[25]:
```python
val_t1_numbers = print_out(val_jsons, 'article_text', 'Validation Data', 1)
bins_list = list(range(0, 200, 10))
plot_graph(bins_list, val_t1_numbers, 'val_t1_1.png')

bins_list = list(range(0, 50, 2))
plot_graph(bins_list, val_t1_numbers, 'val_t1_2.png')
```

```
Validation Data
-------------------------------------------------------------
Number of articles:6436
Longest:270.1077844311377
Shortest:11.319824753559693
Average:29.0
```



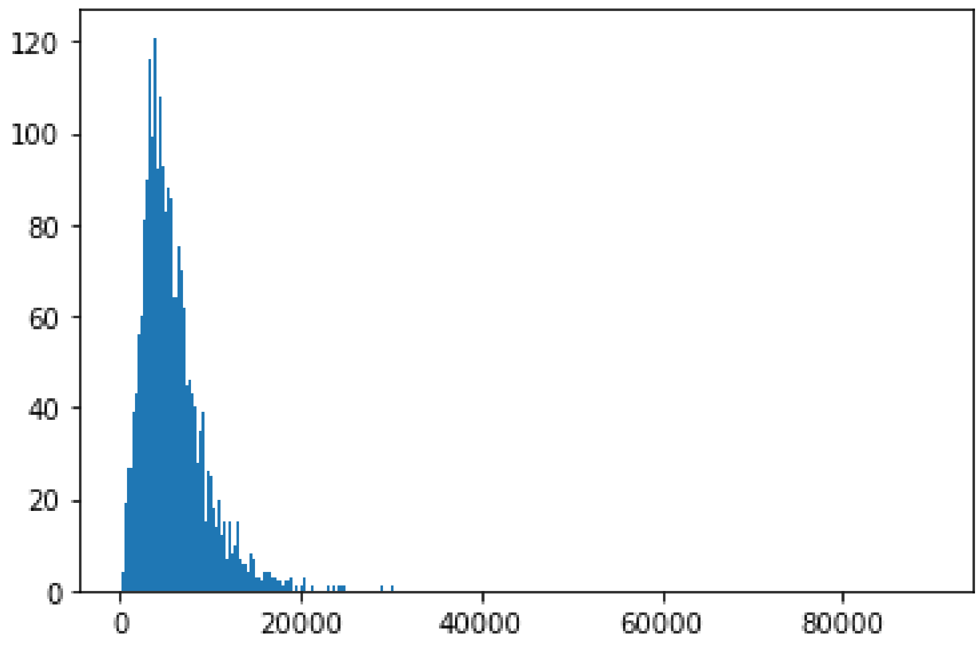


1.3.3 Validation data: number of tokens in an article

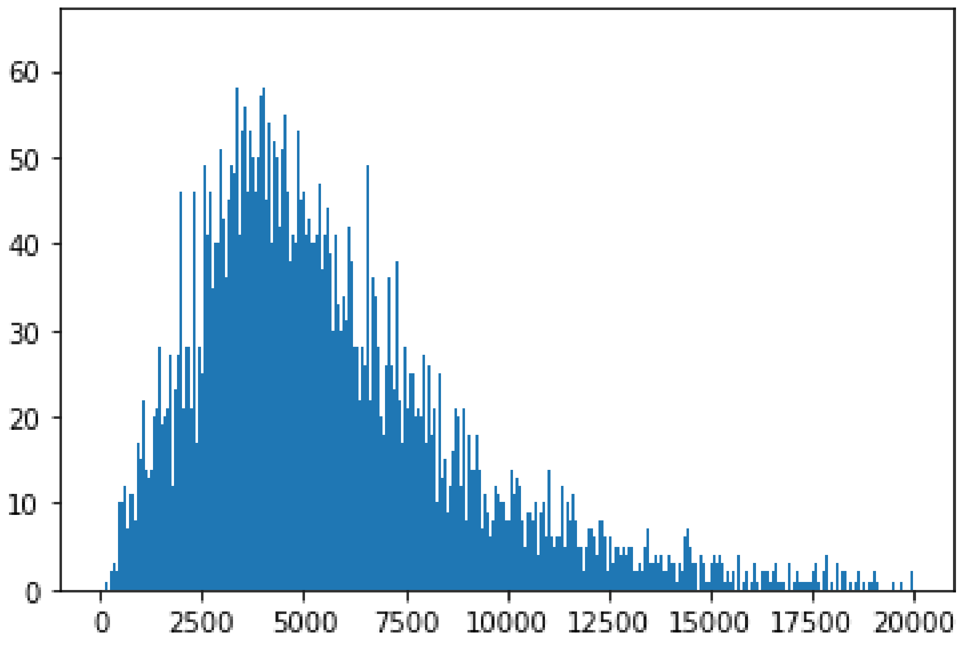
[26]:
```
val_t2_numbers = print_out(val_jsons, 'article_text', 'Validation Data', 2)
bins_list = list(range(100,90000,100))
plot_graph(bins_list, val_t2_numbers, 'val_t2_1.png')

bins_list = list(range(0,20000,50))
plot_graph(bins_list, val_t2_numbers, 'val_t2_2.png')
```

```
Validation Data
-------------------------------------------------------------
Number of articles:6436
Longest:45108
Shortest:195
Average:5894
```



120
100
80
60
40
20
0
0
20000
40000
60000
80000



60
50
40
30
20
10
0
0
2500
5000
7500
10000
12500
15000
17500
20000

1.4 Train+Validation+Test

```
[27]: jsons = train_jsons + val_jsons + test_jsons
```

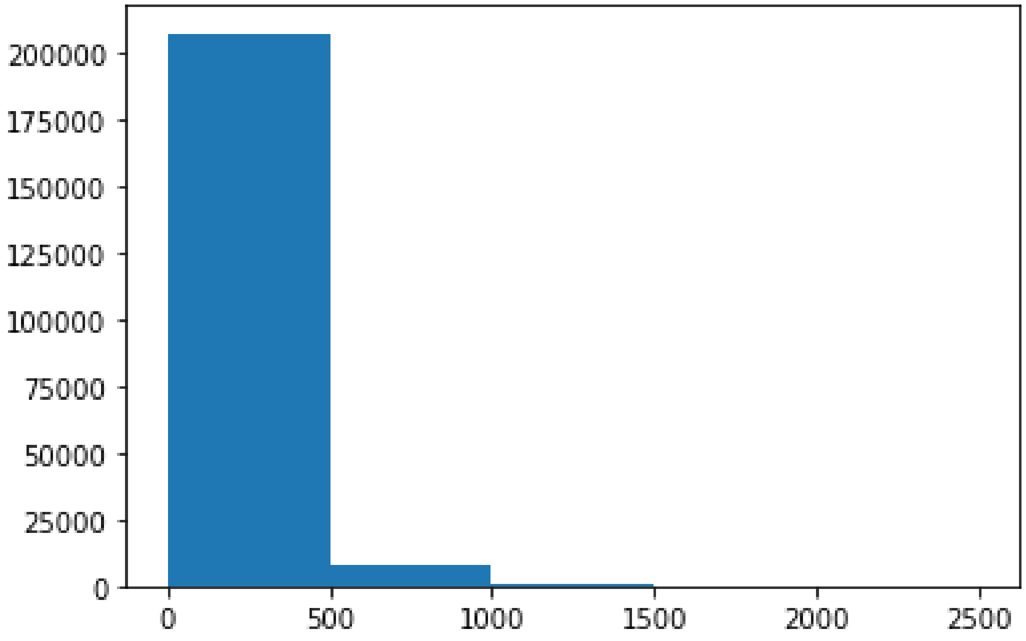
1.4.1 number of sentences in an article

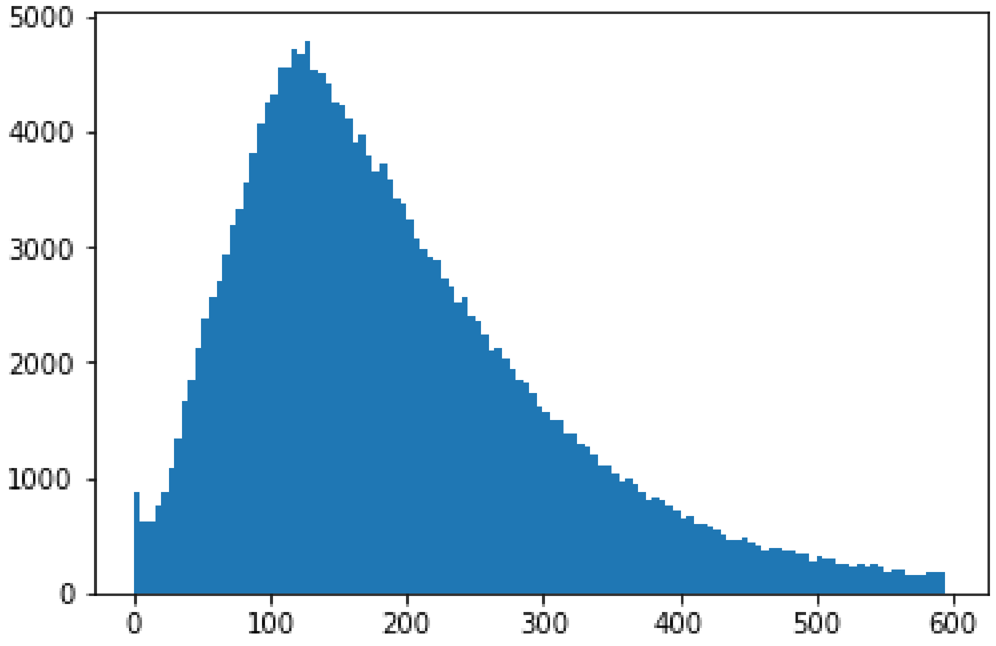
```
[28]: s_numbers = print_out(jsons, 'article_text', 'Validation Data', 3)
```

```
Validation Data
-------------------------------------------------------------
Number of articles:215913
Longest:4615
Shortest:1
Average:206
```

```
[29]: bins_list = list(range(0,3000,500))
      plot_graph(bins_list, s_numbers, 's_1.png')

      bins_list = list(range(0,600,5))
      plot_graph(bins_list, s_numbers, 's_2.png')
```



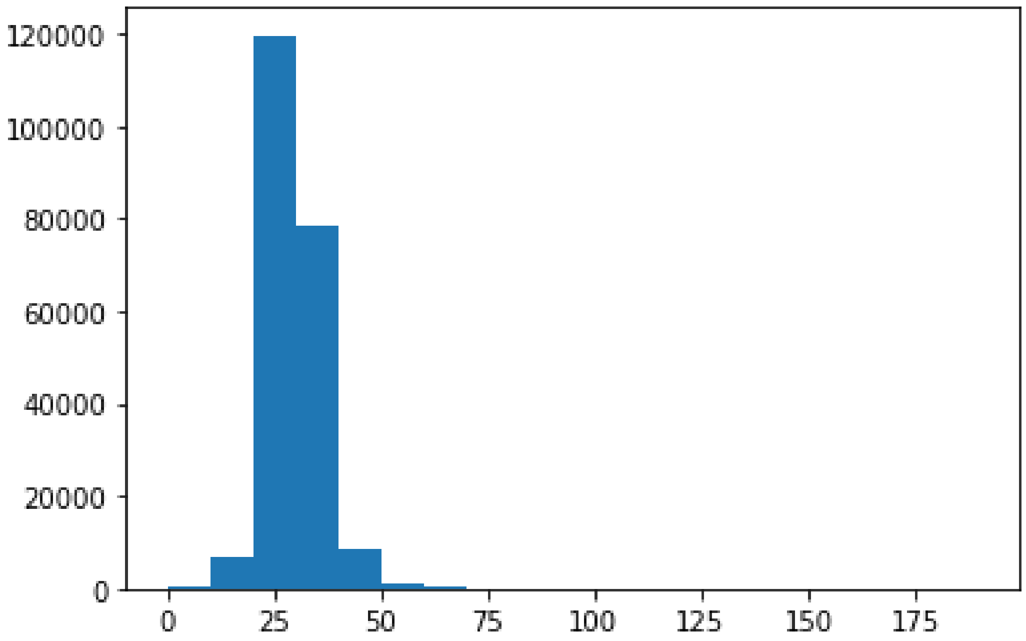


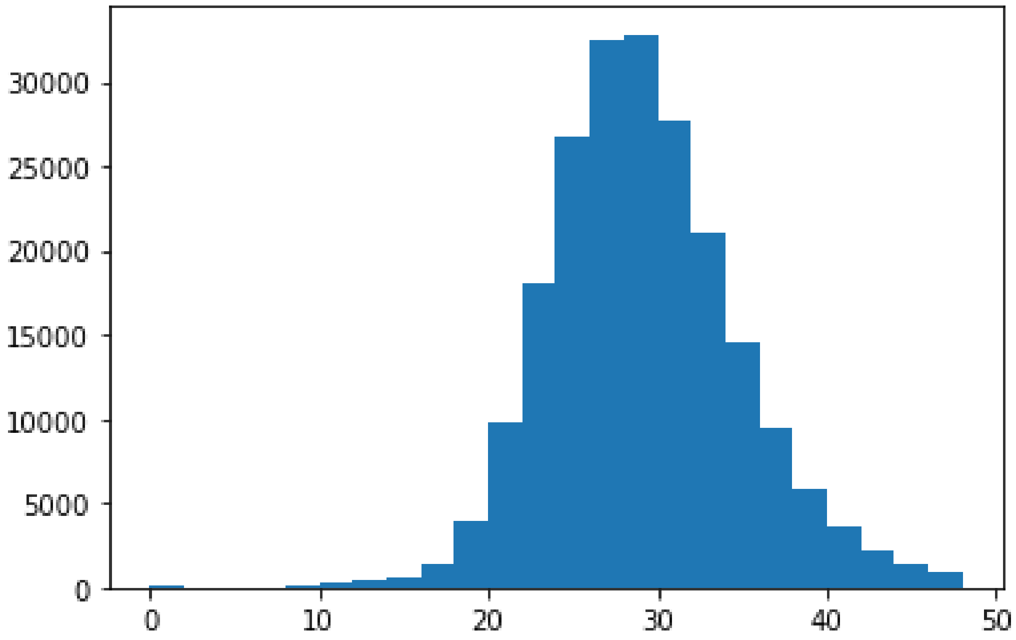
1.4.2 number of tokens in a sentence

```
[30]: t1_numbers = print_out(jsons, 'article_text', 'Validation Data', 1)
bins_list = list(range(0, 200, 10))
plot_graph(bins_list, t1_numbers, 't1_1.png')

bins_list = list(range(0, 50, 2))
plot_graph(bins_list, t1_numbers, 't1_2.png')
```

```
Validation Data
------------------------------------------------------------
Number of articles:215913
Longest:2317.0
Shortest:0.0
Average:29.0
```


0
20000
40000
60000
80000
100000
120000
0
25
50
75
100
125
150
175

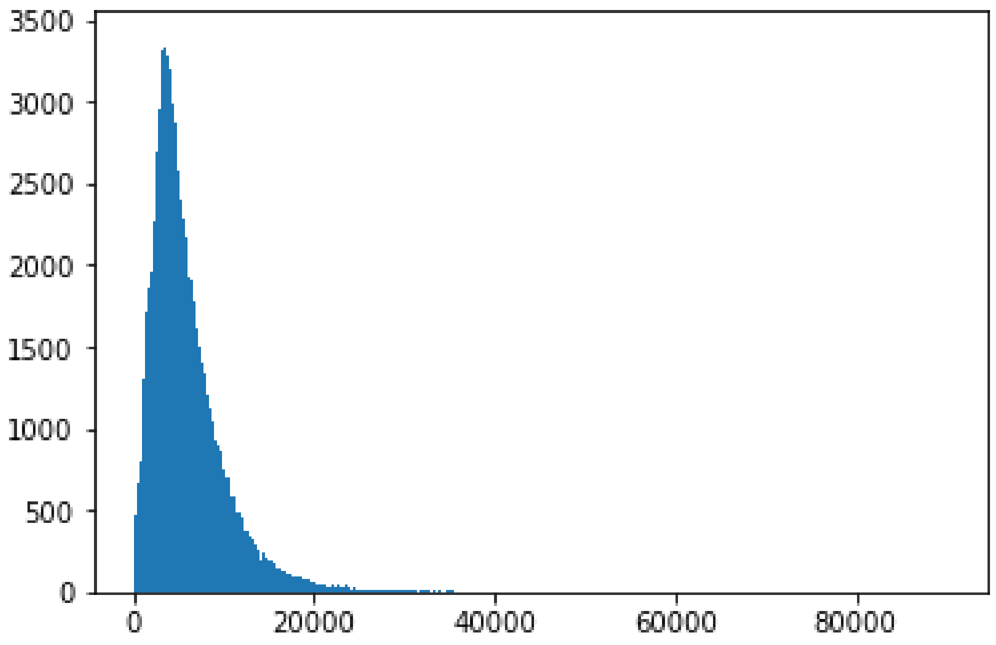

0
5000
10000
15000
20000
25000
30000
0
10
20
30
40
50

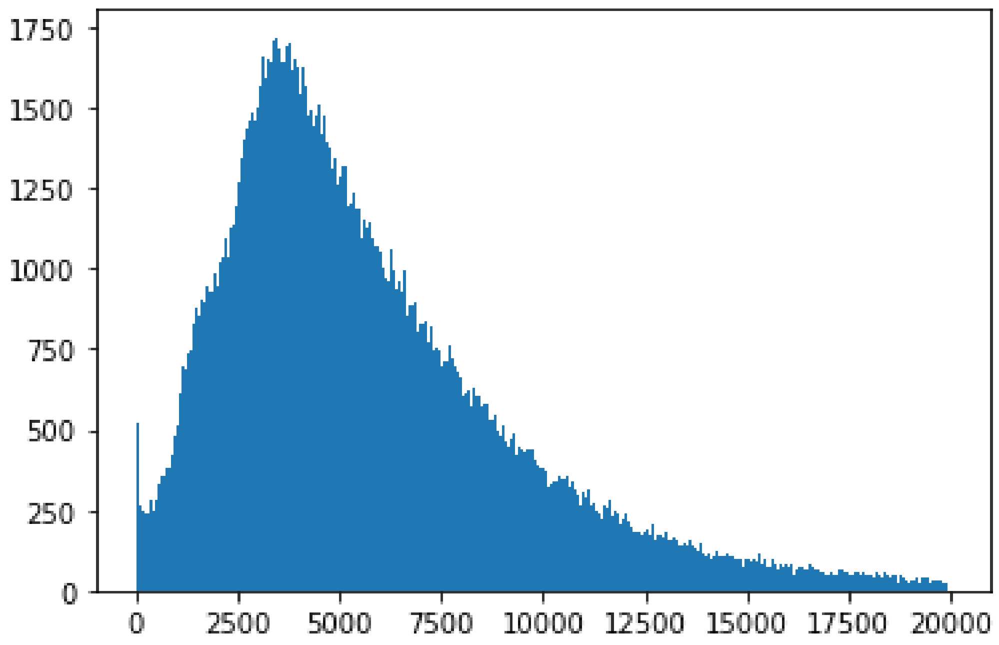
1.4.3 number of tokens in an article

[31]:
```
t2_numbers = print_out(jsons, 'article_text', 'Validation Data', 2)
bins_list = list(range(100,90000,100))
plot_graph(bins_list, t2_numbers, 't2_1.png')

bins_list = list(range(0,20000,50))
plot_graph(bins_list, t2_numbers, 't2_2.png')
```

```
Validation Data
-------------------------------------------------------------
Number of articles:215913
Longest:157180
Shortest:0
Average:6029
```





1750
1500
1250
1000
750
500
250
0
0
2500
5000
7500
10000
12500
15000
17500
20000